ARRAY

TYPES OF ARRAY

- It is a group of variables of similar data types referred by single element.
- It elements are stored in a contiguous memory location.
- The size of array should be mentioned while declaring it.
- Array elements are always counted from zero (0) onward.
- Array elements can be accessed using the position of the element in the array.
- Array can have one or more dimensions.

Advantages of an Array in C

- Random access of elements using array index.
- Use of fewer line of code as it creates a single array of multiple elements.
- Easy access to all the elements.
- Traversal through the array becomes easy using a single loop.
- Sorting becomes easy as it can be accomplished by writing fewer line of code.

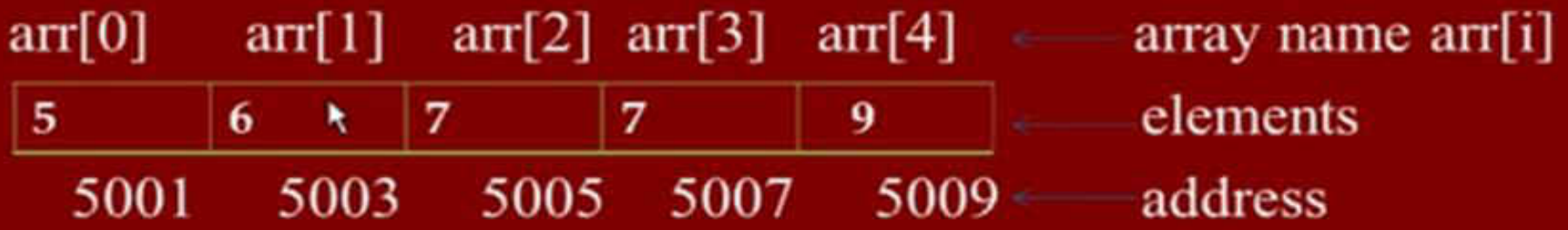
TYPES OF ARRAY

TYPES OF ARRAY

1. One Dimension Array
2. Two Dimension Array
3. Three Dimension Array

One Dimension Array

Ex:Arr[5] ={1 row, multi column}



Declaration:

<data-type> <array name> [size];

EX: int arr[5];

Two Dimension Array

ex: arr[i][j] ={multi row, multi column};

arr[][j] column

arr[i][] row

Arr[0][0]	Arr[0][1]	Arr[0][2]	Arr[0][3]	Arr[0][4]
Arr[1][0]	Arr[1][1]	Arr[1][2]	Arr[1][3]	Arr[1][4]
Arr[2][0]	Arr[2][1]	Arr[2][2]	Arr[2][3]	Arr[2][4]
Arr[3][0]	Arr[3][1]	Arr[3][2]	Arr[3][3]	Arr[3][4]

Declaration:

<data-type> <array_nm> [row_subscript][column-subscript];

Ex: int arr[4][5];

Three Dimension Array

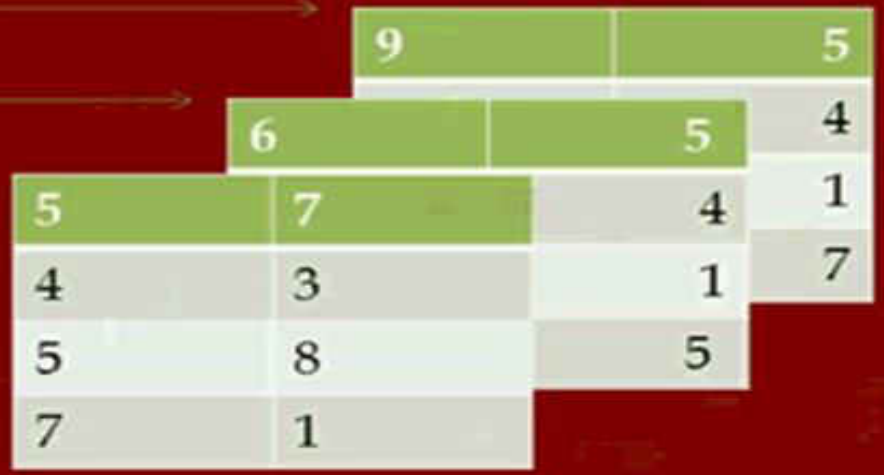
Ex: arr[i][j][k] = arr[3][4][2];

array of arrays of arrays

2nd 2D Array

1st 2D Array

0th 2D Array



Declaration:

<data-type> <array nm>[array subscript][row subscript][column subscript];

Ex: float arr[3][4][2]

One Dimension Char Array

```
void main()
{
char a[5]=“HELLO”;
printf(" %c",a[3]);
}
```

One Dimension Char Array

```
void main()
{
char a[5]=“HELLO”;
printf(" %c",a[0]);
printf(" %c",a[1]);
printf(" %c",a[2]);
printf(" %c",a[3]);
printf(" %c",a[4]);
}
```

One Dimension Char Array

```
void main()
{
int i;
char a[5]=“HELLO”;
for(i=0;i<5;i++)
{
printf(" %c",a[i]);
}
```

One Dimension int Array

```
void main()
{
int a[5]= {45, 54, 46, 76, 38};
printf(" %d",a[0]);
printf(" %d",a[1]);
printf(" %d",a[2]);
printf(" %d",a[3]);
printf(" %d",a[4]);
}
```

One Dimension int Array

```
void main()
{
int i;
int a[5]= {45, 54, 46, 76, 38};
        for(i=0;i<5;i++)
        {
        printf(" %d",a[i]);
        }
}
```

One Dimension Array

```
void main()
{
int a[5],i;
printf(“ENTER MARKS OF 5 SUBJECTS");
for(i=0;i<5;i++)
{
scanf("%d",&a[i]);
}
printf(“ENTERED MARKS ARE AS FOLLOWS : \n");
for(i=0;i<5;i++)
{
printf(" %d /t",a[i]);
} }
```

2D ARRAY

Two Dimension Array

The array which is used to represent and store data in a tabular form is called as 'two dimensional array.' Such type of array specially used to represent data in a matrix form.

Example:

```
int a[3][3] = {2, 3, 5,5,6,8,4,5};
char ch[4][8] = "TechnoExam indelhi " ;
float stax[2][2] = {5003.23, 1940.32, 123.20,-45.5} ;
```

Syntax:

```
<data-type> <array_nm> [row_subscript][column-subscript];
```

Total Size (in Bytes):

Ex: int a[3][3];

'a' is an array of type integer which has storage size of 3 * 3 matrix.

Total size = length of array * size of data type

Ex :3 * 3 * 2 = 18 bytes.

Two Dimension Array

	Column 1	Column 2	Column 3	Column 4
Row 1	x[0][0]	x[0][1]	x[0][2]	x[0][3]
Row 2	x[1][0]	x[1][1]	x[1][2]	x[1][3]
Row 3	x[2][0]	x[2][1]	x[2][2]	x[2][3]

Different ways to initialize two dimensional array

```
int a[2][3] = {{1, 3, 0}, {-1, 5, 9}};

int a[][3] = {{1, 3, 0}, {-1, 5, 9}};

int a[][] = {1, 3, 0, -1, 5, 9};
```

Example - Two Dimension Array

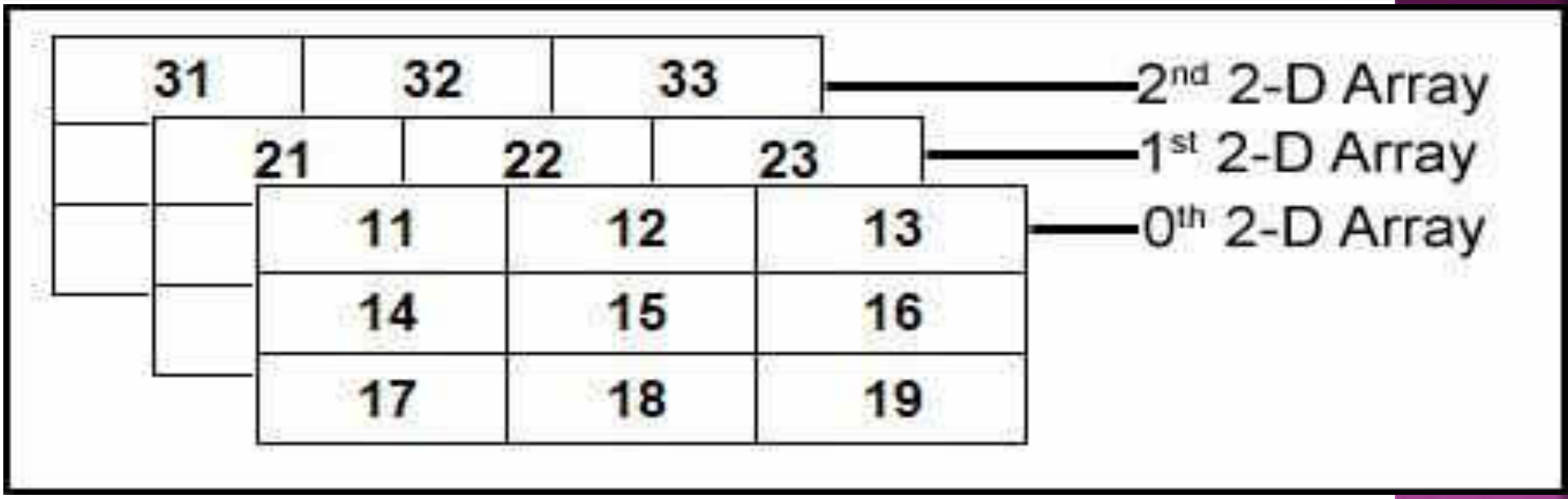
```
#include<stdio.h>
#include<conio.h>
void main()
{
int arr[3][3];
int i,j;
clrscr();
printf(“enter the 9 number:");
for (i=0;i<3;i++)
{
for(j=0;j<3;j++)
{
scanf("%d",&arr[i][j]);
} }
printf("number are as follows:\n");
{
for (i=0;i<3;i++)
    {
     for (j=0;j<3;j++)
          {
          printf("\t%d",arr[i][j]);
          }
printf("\n");
}
getch();
}}
```

3D ARRAY

Three Dimension Array

- ✓A 3D array is essentially an array of arrays of arrays:
- ✓it's an array or collection of 2D arrays,
- ✓and a 2D array is an array of 1D array.
- ✓It may sound a bit confusing, but as you practice working with multidimensional arrays, you start to grasp the logic.
- ✓The next diagram will help you to understand:

Three Dimension Array



<---------- 0th 2D Array ---------->									<---------- 1st 2D Array ---------->									<---------- 2nd 2D Array ---------->								
11	12	13	14	15	16	17	18	19	21	22	23	24	25	26	27	28	29	31	32	33	34	35	36	37	38	39
1000	1002	1004	1006	1008	1010	1012	1014	1016	1018	1020	1024	1026	1028	1030	1032	1034	1036	1038	1040	1042	1044	1046	1048	1050	1052	1054

Example - Three Dimension Array

```
#include<stdio.h>
#include<conio.h>
 void main()
{
int i, j, k;
int arr[3][3][3]=
      {
         {
         {11, 12, 13},
         {14, 15, 16},
         {17, 18, 19}
      },
      {
                {21, 22, 23},
                {24, 25, 26},
                {27, 28, 29}
      },
      {
                {31, 32, 33},
                {34, 35, 36},
                {37, 38, 39}
      },
      };
```

Three Dimension Array

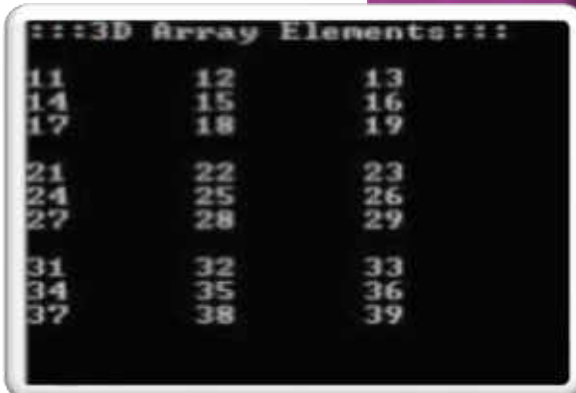
```
clrscr();
printf(":::3D Array Elements:::\n\n");
for(i=0;i<3;i++)
{
        for(j=0;j<3;j++)
                {
                for(k=0;k<3;k++)
                        {
                        printf("%d\t",arr[i][j][k]);
                        }
                printf("\n");
                }
   printf("\n");
}  getch();  }
```

Example - Three Dimension Array

```
void main()
{
int i,j,k;
int  arr[3][3][3];
printf("Enter 3D array values:");
for(i=0;i<3;i++)
{
for(j=0;j<3;j++)
{
for(k=0;k<3;k++)
{
scanf("%d",&arr[i][j][k]);
}}}
```

Three Dimension Array

```
clrscr();
printf(":::3D Array Elements:::\n\n");
for(i=0;i<3;i++)
{
        for(j=0;j<3;j++)
                {
                for(k=0;k<3;k++)
                        {
                        printf("%d\t",arr[i][j][k]);
                        }
                printf("\n");
                }
   printf("\n");
}  getch();  }
```



ARRAY with FUNCTION

```
#include<iostream.h>
#include<conio.h>
void arr(int);
void main()
{
 int i;
 clrscr();
 int marks[]={23,34,45,62,78};
 for(i=0;i<5;i++)
 {
 // cout<<marks[i]<<endl;
 arr(marks[i]);
 }
 getch(); }
```

```
void arr(int m)
{
 cout<<m<<endl;
 }
```

Example – Array with Function

```
#include<stdio.h>
#include<conio.h>
void display(int);
void main()
{
int i;
int marks[]={23,34,45,62,78};
clrscr();
for(i=0;i<5;i++)
{
display(marks[i]);
}
getch();
}
```

```
void  display(int m)
{
printf("\n\n%d",m);
printf("\t modified array is %d:",m+2);
}
```

ARRAY with POINTER

```
void main()
{
int p, arr[ ] = {2,3,4,5,6};
printf(“array elements are as follows:");
for(p=0;p<5;p++)
{
printf(" \n %d and address of arrays: %u",arr[p],&arr[p]);
}
}
```

THANKS